

**Pharmacometrics Markup Language (PharML)  
Level 1, Version 1**

**Standards for system-to-system interchange of data, models and  
related metadata for Pharmacometric analyses**

**Prepared by  
Novartis Pharma AG, Novo Nordisk A/S, Servier & Uppsala University**

## Table of Contents

1	Introduction .....	5
1.1	Scope of PharML Level 1 .....	5
1.2	Limitation of scope for Version 1 of PharML Level 1 .....	6
1.3	Requirements for the NLMEc platform and objective of PharML Level 1 .....	7
1.4	General information on XML.....	8
2	Overview of PharML .....	8
3	Generic mathematical model representation .....	10
3.1	Definitions of indices, dimensions, and variables.....	10
3.2	Dosing history .....	13
3.3	Parameter model .....	13
3.4	Prior distribution.....	14
3.5	Optional algebraic equations.....	15
3.6	Differential equations.....	15
3.7	Observation equations .....	16
3.7.1	Continuous data analysis.....	16
3.7.2	Categorical data analysis.....	17
3.8	Mixture models.....	18
4	PharML components.....	19
4.1	Analysis .....	19
4.1.1	AnalysisStep .....	19
5	Perspectives .....	45
5.1	Future enhancements to PharML: Level 2 and beyond.....	45
5.2	Relationship to other efforts.....	45
5.3	Availability.....	45
6	References .....	46

## Glossary and abbreviations

Abbreviation	Description
<b>AUC</b>	Area Under the Curve
<b>BUGS</b>	Bayesian inference Using Gibbs Sampling; software application
<b>CDISC</b>	Clinical Data Interchange Standards Consortium
<b>CeIIML</b>	A language for storing and exchanging computer-based mathematical models
<b>CL</b>	Clearance
<b>CR</b>	Carriage Return
<b>CRLF</b>	Carriage Return/Line Feed
<b>EBE</b>	Empirical Bayes Estimate
<b>FO</b>	First Order
<b>FOCE</b>	First Order (Conditional Estimation)
<b>FOCEI</b>	First Order (Conditional Estimation) with Interaction
<b>IIV</b>	Interindividual variability
<b>IOV</b>	Interoccasional variability
<b>K<sub>10</sub></b>	Rate constant of elimination
<b>LF</b>	Line Feed
<b>LOQ</b>	Limit Of Quantification
<b>MathML</b>	Mathematical Markup Language
<b>MATLAB</b>	Mathematical modelling environment
<b>MONOLIX</b>	MOdèles NON Linéaires à effets miXtes (NONLinear Mixed Effects Modelling); software application
<b>NLMEc</b>	Nonlinear Mixed Effects Consortium
<b>NONMEM</b>	NONLinear Mixed Effects Modelling; software application
<b>ODE</b>	Ordinary Differential Equation
<b>PD</b>	Pharmacodynamic(s)
<b>PharML</b>	Pharmacometrics Markup Language
<b>PK</b>	Pharmacokinetic(s)
<b>R</b>	Statistical programming environment
<b>S-ADAPT</b>	Computational model platform developed for PK and PD applications

<b>Abbreviation</b>	<b>Description</b>
<b>S-PLUS</b>	Statistical programming environment
<b>SAEM</b>	Stochastic Approximation Expectation Maximization
<b>SAS</b>	Statistical Analysis System
<b>SBML</b>	Systems Biology Markup Language
<b>SDE</b>	Stochastic Differential Equation
<b>SQL</b>	Structured Query Language
<b>SVG</b>	Scalable Vector Graphics
<b>URL</b>	Universal Resource Locator
<b>V</b>	Volume of distribution
<b>XML</b>	eXtensible Markup Language

## 1 Introduction

This document introduces the **Pharmacometrics Markup Language (PharML) Level 1, Version 1** – a description language for data, models and related metadata for pharmacometric analyses (e.g. population PK and/or PD analyses, disease progression modelling, ligand binding analyses, etc.). PharML is developed and maintained by the Nonlinear Mixed Effects Consortium (NLMEc). The objective of the NLMEc is to enable and facilitate the development of a technical implementation of a novel modelling and simulation software platform. This platform is based on an interoperability framework that enables the consolidation of independent plug-ins onto the platform, and PharML is intended to be the open standard format for data, models and related metadata that facilitates this.

PharML objects will be encoded using XML, the eXtensible Markup Language.

### 1.1 Scope of PharML Level 1

The platform that the NLMEc is seeking to develop should ultimately support the entire workflow shown in Figure 1, i.e. all three of the major modules – Design, Analysis, and Interpretation – as they apply to standard pharmacometric modelling and simulation of PK and PD. More specifically, the platform should provide an interoperability framework that binds together independent plug-ins across these modules as well as within each of the modules. This is illustrated for the Analysis module in Figure 2.

A key step in the development of such an integrated modelling environment is the definition of standards for the system-to-system interchange of data, models and related metadata. PharML Level 1 is intended to provide this.

**NOTE:** Being a standard language for system-to-system interchange of data, models and related metadata, PharML Level 1 is not intended to be easily human-readable. A related standard language for user-to-system specification of data and models based on PharML Level 1 may be developed at a later stage, but is currently not considered here.

## 1.2 Limitation of scope for Version 1 of PharML Level 1

The scope of this first version of PharML Level 1 is limited to the Analysis module of the platform, but in future versions it will be extended to cover Design and Interpretation as well.

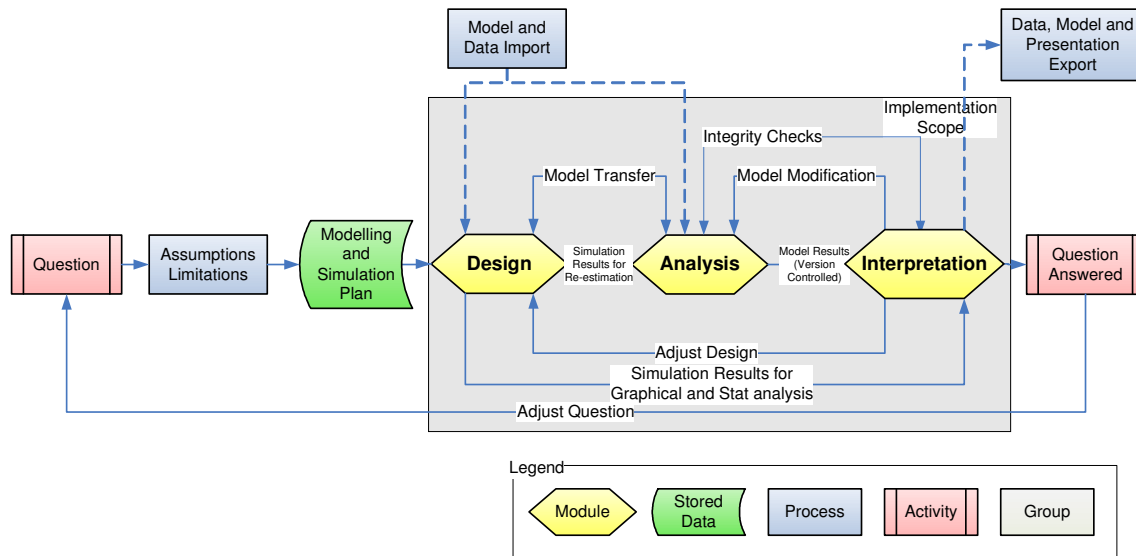


Figure 1: The modelling & simulation workflow.

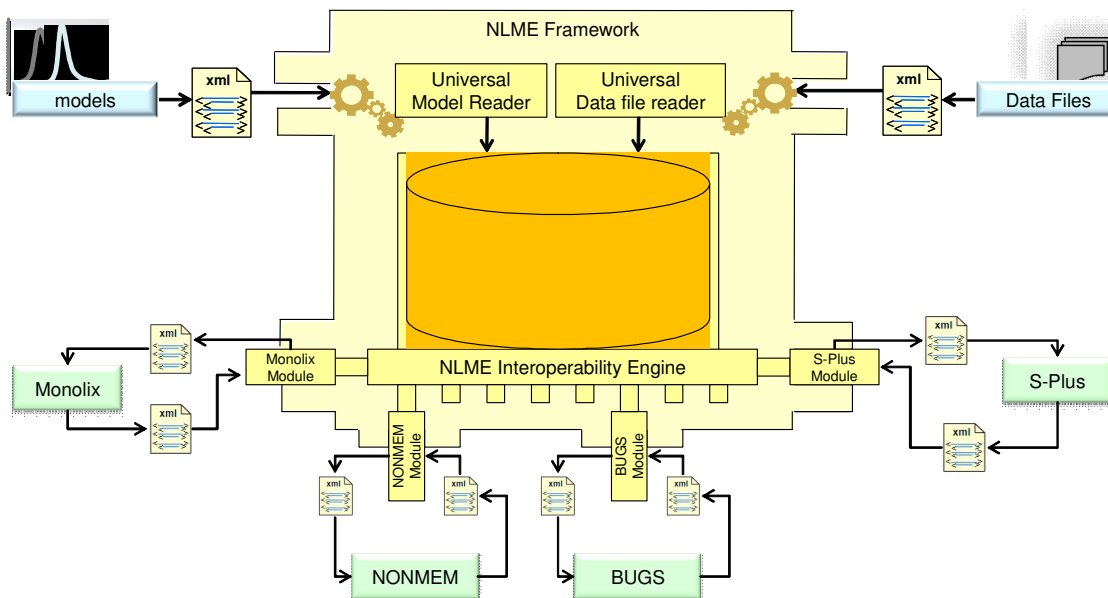


Figure 2: Concept of the interoperability framework.

### 1.3 Requirements for the NLMEc platform and objective of PharML Level 1

Figure 2 illustrates the interoperability of the platform for the Analysis module. More specifically, it illustrates that the system providing the interoperability should be able to:

- Read application-independent data and model files in XML format (PharML).
  - A model file in XML format could be obtained by translating an application-specific model file (e.g. a NONMEM control stream) into XML, or it might be created *de novo* through a (graphical) user interface, e.g. via a standard user-to-system language for specification of models. This point is currently out of scope of the present document – cf. section 1.1.
- Translate the XML-formatted data and model files into application-specific data and model files for applications such as NONMEM and MONOLIX.
- Run the different applications and translate the program output into XML format.
- Provide summary graphical and tabular reports of the output of the applications based on the XML-formatted output.

The objective of PharML Level 1 is to define the minimum information that should potentially be documented in the XML files to be able to translate these XML files into any application-specific data and model files (for e.g. NONMEM, MONOLIX, BUGS, S-PLUS, R, MATLAB, S-ADAPT, and SAS).

**NOTE:** The work of CDISC – a global, open, multidisciplinary, non-profit effort to define standards to support the acquisition, exchange, submission and archive of clinical research data and metadata – is currently ongoing and expected to set the stage in terms of data formatting standards in the future (see <http://www.cdisc.org>). It is thus the aim of PharML Level 1 to use CDISC standards for data sets, if this is possible. For Version 1 of PharML Level 1, however, the aim is to define the minimum information needed to be able to create the necessary application-specific data files. For the next version of PharML Level 1, it will then be addressed if CDISC offers standards for storing all the information needed, and how to do the mapping between CDISC-standardized data sets and PharML Level 1 data files. If CDISC does not offer standards for storing important parts of the information needed, the

NLMEc should discuss with CDISC, if this information could be included in the standard in future versions (or if PharML could be made a part of CDISC altogether).

#### 1.4 General information on XML

The eXtensible Markup Language (XML, <http://www.w3.org/XML>) is a general-purpose markup language. It is classified as an extensible language because it allows its users to define their own tags. Its primary purpose is to facilitate the sharing of structure data across different information systems. An XML application is a domain-specific language for describing virtually any type of structured information. For example, SVG (<http://www.w3.org/Graphics/SVG>) is an XML application for describing vector graphics; MathML (<http://www.w3.org/Math>) is an XML application for describing mathematical expressions; SBML (<http://www.sbml.org>) is an XML application which embeds the former two, along with other XML applications, to describe models for biochemical reaction networks; CellML (<http://www.cellml.org>) is another open language for describing biological models, but has since grown into wider applicability. XML can be used by most computer languages and can be easily converted to formats that others can read. It is a data storage format, not intended to be human-readable, and provides a storage structure which can be mapped into object-oriented models for code and databases. It may in turn be adapted into relational structures for relational databases. XML has the advantages of simplicity, ubiquity, almost infinite extensibility, interoperability, and openness of its standard and code.

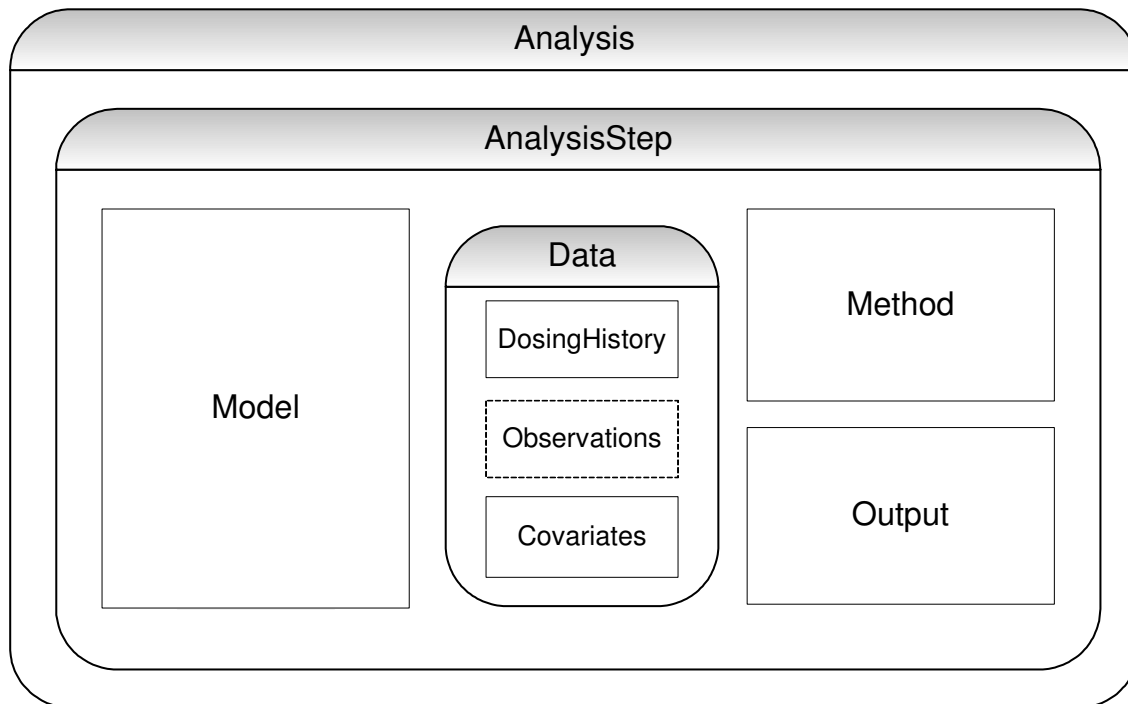
## 2 Overview of PharML

Given the current scope of PharML, where the aim is to define a standard for system-to-system interchange of data, models, and related metadata to support the Analysis module of the overall framework, it is obvious to name the highest level component in an object-oriented definition of this standard **Analysis**. An **Analysis** would then consist of a number of **AnalysisSteps**, which would contain a **Model** definition, the **Data** used to estimate the parameters of the model, the **Method** used for the estimation, and some **Output** describing the model and how well it fits the data. This is illustrated in Figure 3.

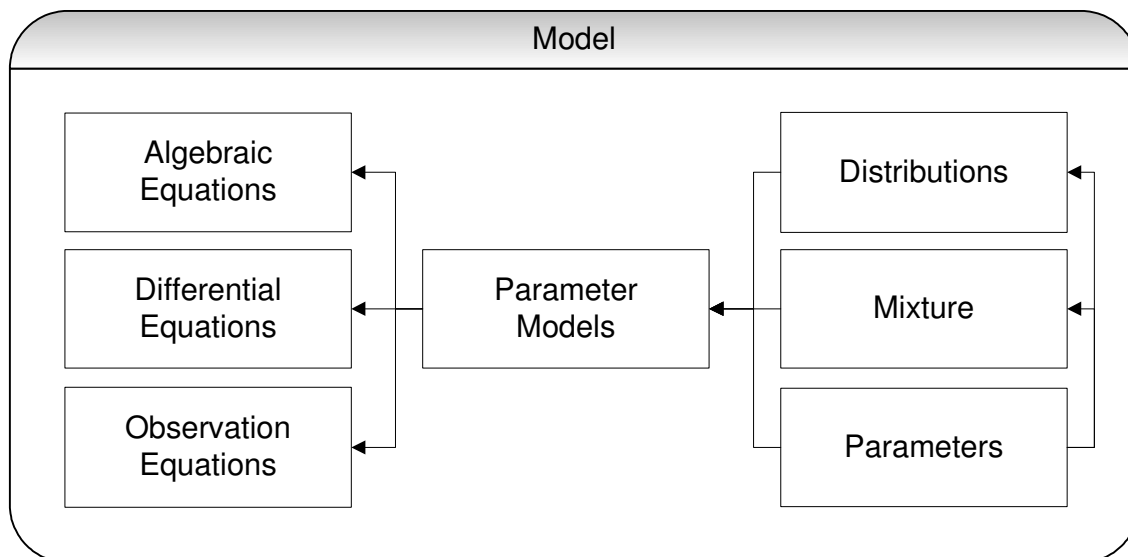
**NOTE:** Bootstrapping and related resampling procedures for model evaluation are considered a part of the Interpretation module of the overall framework, and are thus not discussed here. It is clear, however, that in order to properly deal with such

procedures, it will be necessary to introduce support for this type of analysis as a part of **AnalysisStep** or elsewhere.

A representation of the **Model** object and its subcomponents is given in Figure 4. More detailed descriptions of the individual components (**Analysis**, **AnalysisStep**, **Data**, **Model**, **Method**, **Output**) and their subcomponents are given in section 4.



**Figure 3: Diagrammatic representation of an Analysis object. It was decided to store DosingHistory, Observations and Covariates as separate objects, because this is how clinical data are usually stored and because it will be necessary to be able to change these independently of each other, when performing simulations (in the Design module).**



**Figure 4: Diagrammatic representation of a Model object.**

### 3 Generic mathematical model representation

The definition of PharML (primarily the definition of **Model** and its subcomponents) is based on a generic mathematical model representation, which is outlined in the following.

**NOTE:** The generic model is written for two levels of nested variability: inter-individual variability (IIV) and inter-occasion-within-subject variability (IOV), but it can easily be generalized even further to include more levels of nested variability.

**NOTE:** All items defined below are introduced for the purpose of outlining the generic mathematical model representation, and should not be interpreted as notations describing the language in which the user will eventually interact with the system. This has yet to be defined.

#### 3.1 Definitions of indices, dimensions, and variables

- $c$  : Compartment index,  $1 \leq c \leq n_c$ .
- $i$  : Subject index,  $1 \leq i \leq N$ .
- $j$  : Index for observation time points for subject  $i$  on occasion  $k$ ,  $1 \leq j \leq n_{o,ik}$ .
- $k$  : Occasion index for subject  $i$ ,  $1 \leq k \leq m_i$ .

- $l$  : Index for dosing time points for subject  $i$  on occasion  $k$ ,  $1 \leq l \leq n_{d,ik}$ .  
 $p$  : Parameter index,  $1 \leq p \leq n_p$ .  
 $q$  : Covariate index,  $1 \leq q \leq n_{cov}$ .  
 $r$  : Response variable index,  $1 \leq r \leq n_r$ .  
 $s$  : Subpopulation index,  $1 \leq s \leq n_s$ .
- $N$  : Number of subjects.  
 $m_i$  : Number of occasions within subject  $i$ .  
 $n_{d,ik}$  : Number of dosing time points for subject  $i$  on occasion  $k$ .  
 $n_{o,ik}$  : Number of observation time points for subject  $i$  on occasion  $k$ .  
 $n_c$  : Number of compartments.  
 $n_p$  : Number of parameters.  
 $n_{cov}$  : Number of covariates.  
 $n_r$  : Number of response variables.  
 $n_s$  : Number of subpopulations.
- $x$  : Independent variable (e.g. time, dose, concentration, ...) – continuous scale.  
 $x_{ik,j}$  : Independent variable (e.g. time, dose, concentration, ...) at observation  $j$  in subject  $i$  on occasion  $k$ .  
 $t$  : Time – continuous scale.  
 $t_{ik,l}^c$  : Time of dose  $l$  to compartment  $c$  in subject  $i$  on occasion  $k$ .
- $\mathbf{A}_{ik}$  : Vector of amounts in all compartments for subject  $i$  on occasion  $k$ .  
 $\mathbf{A}_{ik,j}$  :  $\mathbf{A}_{ik}$  at observation  $j$ .  
 $A_{ik}^c$  : Amount in compartment  $c$  for subject  $i$  on occasion  $k$ .  
 $A_{ik,j}^c$  :  $A_{ik}^c$  at observation  $j$ .
- $\mathbf{D}_{ik}$  : Dosing history for subject  $i$  on occasion  $k$  – all compartments.  
 $D_{ik}^c$  : Dosing history for subject  $i$  on occasion  $k$  – compartment  $c$ .  
 $d_{ik,l}^c$  : Amount given to compartment  $c$  for subject  $i$  on occasion  $k$  at time  $t_{ik,l}^c$ .  
 $z_{ik,l}^c$  : Type of dose - 0 for bolus, 1 for infusion – compartment  $c$ , subject  $i$ , occasion  $k$ , time  $t_{ik,l}^c$ .  
 $\Delta_{ik,l}^c$  : Duration of infusion (if  $z_{ik,l}^c = 1$ ) – compartment  $c$ , subject  $i$ , occasion  $k$ , time  $t_{ik,l}^c$ .

- $\varphi_{ik}$  : Subject-specific parameter vector for subject  $i$  on occasion  $k$ .
- $\theta$  : Vector of fixed effects parameters.
- $\mathbf{Z}_{ik}$  : Vector of covariates for subject  $i$  on occasion  $k$ .
- $\eta_i$  : Vector of random effects for subject  $i$  (inter-individual variability or IIV).
- $\eta_{ik}$  : Vector of random effects for occasion  $k$  within subject  $i$  (inter-occasion-within-subject variability or IOV).
- $\Omega_{IIV}$  : Covariance matrix for random effects for IIV (assuming a normal distribution).
- $\Omega_{IOV}$  : Covariance matrix for random effects for IOV (assuming a normal distribution).
- $P_{IIV}$  : Distribution of the random effects for IIV (generic representation; refers to a parametric/non-parametric theoretical distribution or to an empirical distribution (EBEs)).
- $P_{IOV}$  : Distribution of the random effects for IOV (generic representation; refers to a parametric/non-parametric theoretical distribution or to an empirical distribution (EBEs)).
- $\mathbf{X}$  : A parameter to be estimated, e.g.  $\theta$ ,  $\Omega_{IIV}$ ,  $\Omega_{IOV}$ , or  $\Sigma$  (generic representation).
- $P_{prior}$  : Prior distribution of a parameter to be estimated (generic representation).
- $\bar{\theta}$  : Prior mean of  $\theta$  (assuming a normal distribution).
- $\Gamma$  : Prior covariance matrix of  $\theta$  (assuming a normal distribution).
- $\bar{\Omega}_{IIV}$  : Mode of  $\Omega_{IIV}$  prior (assuming an inverse Wishart distribution).
- $v$  : Degrees of freedom of  $\Omega_{IIV}$  prior (assuming an inverse Wishart distribution).
- $T^r$  : Link function in categorical data analysis
- $U^r$  : Linear or nonlinear function in terms of  $\theta$  and  $\mathbf{Z}$  in categorical data analysis
- $\mathbf{Z}$  : Known information in categorical data analysis (vector or matrix)
- $p_{is}(\theta)$  : Mixture probabilities for  $i$  subjects distributed across  $s$  subpopulations
- $L_i$  : Marginal likelihood of the data for subject  $i$ , in the context of mixture models
- $l_{is}(\eta, \theta)$  : Conditional likelihood of subject  $i$  belonging to subpopulation  $s$

- $\mathbf{y}_{ik,j}$ : Vector of the  $n_r$  observed responses for subject  $i$  on occasion  $k$  at observation  $j$ .
- $\boldsymbol{\varepsilon}_{ik,j}$ : Vector of the  $n_r$  residual errors obtained for the  $n_r$  observed responses for subject  $i$  on occasion  $k$  at observation  $j$ .
- $y_{ik,j}^r$ : Observed response  $r$  for subject  $i$  on occasion  $k$  at observation  $j$ .
- $\varepsilon_{ik,j}^r$ : Residual error for response  $r$  for subject  $i$  on occasion  $k$  at observation  $j$ .
- $\boldsymbol{\Sigma}$  : Residual error covariance matrix (assuming a normal distribution).

### 3.2 Dosing history

For each subject  $i$  on each occasion  $k$ , dosing history is uniquely specified by indicating for each dose: the compartment in which the dose was given ( $c$ ), the time of dosing ( $t_{ik,l}^c$ ), the amount given ( $d_{ik,l}^c$ ), the dose type ( $z_{ik,l}^c = 0$  for bolus,  $z_{ik,l}^c = 1$  for infusion), and its duration ( $\Delta_{ik,l}^c$ , for infusions only). Based on this, one possible general way to write dosing history  $\mathbf{D}_{ik}$  for subject  $i$  on occasion  $k$  is the following:

$$\mathbf{D}_{ik} = \begin{bmatrix} D_{ik}^1(t) \\ \vdots \\ D_{ik}^c(t) \\ \vdots \\ D_{ik}^{n_c}(t) \end{bmatrix}$$

where:

$$D_{ik}^c(t) = \sum_{l=1}^{n_{d,ik}} (1 - z_{ik,l}^c) \cdot \delta(t - t_{ik,l}^c) \cdot d_{ik,l}^c + z_{ik,l}^c \cdot \left( H(t - t_{ik,l}^c) \cdot \frac{d_{ik,l}^c}{\Delta_{ik,l}^c} - H(t - (t_{ik,l}^c + \Delta_{ik,l}^c)) \cdot \frac{d_{ik,l}^c}{\Delta_{ik,l}^c} \right)$$

and where  $\delta(\cdot)$  is the Dirac delta function and  $H(\cdot)$  is the Heaviside step function.

### 3.3 Parameter model

The parameter model describes the dependence of the occasion-within-subject-specific parameters  $\boldsymbol{\varphi}_{ik}$  on the fixed effects  $\boldsymbol{\theta}$ , the covariates  $\mathbf{Z}_{ik}$ , and the random effects ( $\boldsymbol{\eta}_i, \boldsymbol{\eta}_{ik}$ ) of the model – in general matrix-vector notation for all parameters:

$$\boldsymbol{\varphi}_{ik} = \mathbf{H}(\boldsymbol{\theta}, \mathbf{Z}_{ik}, \boldsymbol{\eta}_i, \boldsymbol{\eta}_{ik}) \quad , \quad \boldsymbol{\eta}_i \sim P_{IV} \quad , \quad \boldsymbol{\eta}_{ik} \sim P_{OV}$$

where the distributions  $P_{IV}$  and  $P_{OV}$  can be non-parametric (i.e. discrete or smooth) or parametric. In all cases,  $\boldsymbol{\eta}_i$  and  $\boldsymbol{\eta}_{ik}$  are assumed to be independent.

When  $P_{IIV}$  and  $P_{IOV}$  are parametric distributions, they can be described by a finite number of parameters, which are estimated. In case of normal distributions, the above general equation becomes:

$$\varphi_{ik} = \mathbf{H}(\boldsymbol{\theta}, \mathbf{Z}_{ik}, \boldsymbol{\eta}_i, \boldsymbol{\eta}_{ik}) \quad , \quad \boldsymbol{\eta}_i \sim N(\mathbf{0}, \boldsymbol{\Omega}_{IIV}) \quad , \quad \boldsymbol{\eta}_{ik} \sim N(\mathbf{0}, \boldsymbol{\Omega}_{IOV})$$

Or, equivalently, for each parameter – in scalar notation for parameter  $p$ :

$$\varphi_{ik}^p = H^p(\boldsymbol{\theta}, \mathbf{Z}_{ik}, \eta_i^p, \eta_{ik}^p)$$

In case of exponential IIV and exponential IOV the parameter model will look as follows:

$$\varphi_{ik} = \mathbf{h}(\boldsymbol{\theta}, \mathbf{Z}_{ik}) \cdot \exp(\boldsymbol{\eta}_i) \cdot \exp(\boldsymbol{\eta}_{ik})$$

As mentioned above, the notation used here may be generalized using more levels of nested variability and different assumptions for the random effects distributions.

**NOTE:** The parameter model is uniquely specified by specifying the  $n_p$  elements of vector  $\mathbf{H}$  and the distribution of the random effects by the  $n_p \times (n_p+1) / 2$  – dimensional upper/lower triangles of  $\boldsymbol{\Omega}_{IIV}$  and  $\boldsymbol{\Omega}_{IOV}$ .

### 3.4 Prior distribution

For Bayesian analyses, informative or non-informative priors will need to be specified for all parameters that are estimated, e.g.  $\boldsymbol{\theta}$ ,  $\boldsymbol{\Omega}_{IIV}$ ,  $\boldsymbol{\Omega}_{IOV}$ , and  $\boldsymbol{\Sigma}$ . This can be done as follows:

$$\mathbf{X} \sim P_{prior}$$

where  $P_{prior}$  represents the prior distribution, and  $\mathbf{X}$  is the parameter to be estimated. For  $\boldsymbol{\theta}$ , normal distributions are often used, i.e.:

$$\boldsymbol{\theta} \sim N(\bar{\boldsymbol{\theta}}, \boldsymbol{\Gamma})$$

whereas, for variance components, inverse Wishart distributions are common, e.g.:

$$\boldsymbol{\Omega}_{IIV} \sim W^{-1}(\bar{\boldsymbol{\Omega}}_{IIV}, \nu)$$

**NOTE:** For normal distributions, the prior distribution is uniquely specified by specifying the elements of the mean vector ( $\bar{\boldsymbol{\theta}}$ ) and the upper/lower triangle of the

covariance matrix ( $\Gamma$ ). For inverse Wishart distributions, the prior distribution is uniquely specified by specifying the upper/lower triangle of the mode matrix ( $\bar{\Omega}_{IV}$ ) and the degrees of freedom ( $\nu$ ).

### 3.5 Optional algebraic equations

Optional algebraic equations are defined as optional assignments of the following type:

$$\rho_{ik} = \mathbf{r}(\varphi_{ik}, x, \mathbf{D}_{ik}, A_{ik})$$

**NOTE:** Optional algebraic equations may be used for several purposes, e.g. for definition of time-dependent processes (using conditions/logical expressions involving  $x$  – e.g. to implement lag time or a discrete change in subject parameters), for definition of (secondary) parameters ( $K_{10}=CL/V$ ,  $AUC=Dose/CL$ , etc.), and for simplification of notation (by introducing a shorter name for a potentially complicated expression used several times).

**NOTE:** Optional algebraic equations, as defined here, will not support the definition of models described by DAE systems (differential-algebraic equation systems) with index  $\geq 1$  (e.g. definition of root finders). An extension to address this will be included in a future release.

### 3.6 Differential equations

Differential equations are defined as assignments of the following type – in general matrix-vector notation for all compartments:

$$d\mathbf{A}_{ik} = \mathbf{g}(\varphi_{ik}, \mathbf{A}_{ik}, \rho_{ik}, x, \mathbf{D}_{ik})dx + \boldsymbol{\sigma}_w(\varphi_{ik}, \rho_{ik}, x, \mathbf{D}_{ik})d\boldsymbol{\omega}_x$$

Or, equivalently, for each compartment – in scalar notation for compartment  $c$ :

$$dA_{ik}^c = g^c(\varphi_{ik}, \mathbf{A}_{ik}, \rho_{ik}, x, D_{ik}^c)dx + \sum_{s=1}^{n_c} \sigma_w^{cs}(\varphi_{ik}, \rho_{ik}, x, D_{ik}^c)d\omega_x^s$$

The particular notation used here encompasses ordinary differential equations (ODEs) as well as stochastic differential equations (SDEs). For ODEs  $\boldsymbol{\sigma}_w$  is simply zero.

The corresponding initial conditions are assignments of the following type – in general matrix-vector notation for all compartments:

$$\mathbf{A}_{ik,0} = \mathbf{q}(\varphi_{ik}, \rho_{ik}, \mathbf{D}_{ik})$$

Or, equivalently, for each compartment – in scalar notation for compartment  $c$ :

$$A_{ik,0}^c = q^c(\varphi_{ik}, \rho_{ik}, D_{ik}^c)$$

**NOTE:** The system of ODEs (SDEs) is uniquely specified by specifying the  $n_c$  elements of vector  $\mathbf{g}$  (and the  $n_c \times n_c$  elements of matrix  $\sigma_w$ ) for the right-hand-sides along with the  $n_c$  elements of vector  $\mathbf{q}$  for the initial conditions.

### 3.7 Observation equations

Two cases have to be considered: continuous data analysis, and categorical data analysis. It is also possible to have a joint model for analysis of both continuous and categorical data.

#### 3.7.1 Continuous data analysis

In the case of continuous data analysis, observation equations are defined as assignments of the following type – in general matrix-vector notation for all response variables:

$$\mathbf{y}_{ik,j} = \mathbf{f}(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}) + \mathbf{F}(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}, \boldsymbol{\varepsilon}_{ik,j}), \quad \boldsymbol{\varepsilon}_{ik,j} \sim N(\mathbf{0}, \Sigma)$$

$$\text{with: } E\{\mathbf{y}_{ik,j}\} = \mathbf{f}(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j})$$

Or, equivalently, for each response variable – in scalar notation for response variable  $r$ :

$$y_{ik,j}^r = f^r(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}) + F^r(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}, \varepsilon_{ik,j}^r)$$

$$\text{with: } E\{y_{ik,j}^r\} = f^r(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j})$$

For compartmental models,  $f^r$  describes the relationship between the amounts in the various compartments and the expected value of the observations (e.g. plasma concentrations) for a given response  $r$ . For pure dose-response models, and for common pharmacokinetic compartmental models, where an explicit solution is available as a function of the independent variables (e.g. time and dosing regimen), the entire structural model is given by this(these) equation(s).

The residual error model,  $\mathbf{F}$ , describes the statistical properties assumed for the residuals. In case of multiplicative error, the residual error model will look as follows:

$$\mathbf{F}(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}, \boldsymbol{\varepsilon}_{ik,j}) = \mathbf{f}(\varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}) \cdot \boldsymbol{\varepsilon}_{ik,j}$$

The notation used here may also be generalized using different distributional assumptions. Non-parametric distributions (given some constraints) can even be considered.

**NOTE:** The observation equations are uniquely specified by specifying the  $n_r$  elements of vector  $\mathbf{f}$ , the  $n_r$  elements of vector  $\mathbf{F}$  and the  $n_r \times (n_r + 1) / 2$  – dimensional upper/lower triangle of  $\boldsymbol{\Sigma}$ .

### 3.7.2 Categorical data analysis

Categorical data can be dichotomous (two categories e.g. success/failure), polychotomous (multiple categories, e.g. complete/partial/no response) with or without inherent order. They can also refer to a series of events: serial discrete events (e.g. seizures), or single event (e.g. survival).

For each response type  $r$ , the observations  $y^r$  are assumed to be generated from a particular distribution function in the exponential family (which includes most of the common distributions e.g. normal, binomial, Poisson distributions...).

The expected value of  $y^r$  (i.e. the mean of the distribution) can be defined using generalized linear models or non linear models as assignments of the following type:

$$E\{y^r\} = T^{r-1}(U^r(\boldsymbol{\theta}, \mathbf{Z}))$$

where  $T^r$  is the link function and depends on the type of categorical data studied (e.g. for dichotomous data, the logit function is typically used as the link function).  $U^r$  is a linear (or non-linear) function of  $\boldsymbol{\theta}$  and  $\mathbf{Z}$ , where  $\mathbf{Z}$  refers to known information (e.g. documented covariates, doses administered, etc.).

Similar models can be proposed at the subject level (for example, it can be assumed that for each subject, the dependent variable follows a Bernoulli distribution). The

expected value of  $y_{ik}^r$  can then be defined as a function of  $\theta$  and  $\mathbf{Z}_{ik}$  in a deterministic way:

$$E\{y_{ik}^r\} = T^{r-1}(U^r(\theta, \mathbf{Z}_{ik}))$$

The above equation can be further generalized in order to include additional explanatory factors (e.g. predicted concentration, predicted response, time...)

$$E\{y_{ik,j}^r / \boldsymbol{\eta}_i, \boldsymbol{\eta}_{ik}\} = T^{r-1}(U^r(\theta, \mathbf{Z}_{ik}, \varphi_{ik}, \mathbf{A}_{ik,j}, \mathbf{D}_{ik}, \rho_{ik}, x_{ik,j}))$$

**NOTE:** The above models can be used to perform survival analysis using particular survival distributions (e.g. exponential, Weibull, Gompertz).

### 3.8 Mixture models

In cases in which the PK or PD of a drug are found to be polymorphic within a population, mixtures of probability distributions may be used to model a probability distribution (Frame, 2007). Mixtures of this kind can be biological, as for (unknown) acetylator phenotype in the example of isoniazid, or statistical, which might be needed to overcome a situation in which a skewed random-effects variable cannot be made symmetric by adjustments to the structural model or by transformation of the random effect.

$$L_i = \sum_{s=1}^{n_s} p_{is}(\theta) \cdot \int l_{is}(\eta, \theta) \cdot h(\eta, \Omega) d\eta$$

When a separate random effect is used for a second subpopulation, the marginal likelihood of the data for subject  $i$  is given by  $L_i$ , the average of the marginal likelihoods over  $n_s$  subpopulations (or models). Mixture probability  $p_{is}(\theta)$  varies between the  $n_s$  subpopulations and the  $N$  subjects. This expression suggests that  $\eta$  can assume different values for the different subpopulations. The term  $h(\eta, \Omega)$  is the density function for  $\eta$ ,  $l_{is}(\eta, \theta)$  is the conditional likelihood, and the Empirical Posterior Probability ( $EPP_i$ ) that subject  $i$ 's data is described by submodel  $s$  is given by

$$\frac{p_{is}(\theta) \cdot \int l_{is}(\eta, \theta) \cdot h(\eta, \Omega) d\eta}{L_i} = EPP_i$$

The value of  $s$  corresponding to the largest of these  $EPP_i$  values is typically taken to be the subpopulation to which subject  $i$  belongs.

Mixture probabilities may be static or dynamic, and may be computed in the context of models for both likelihood or observation predictions.

## 4 PharML components

In this section we define each of the major structures in PharML.

### 4.1 Analysis

The **Analysis** is the highest-level component in PharML. Only one component of type **Analysis** is allowed in a PharML document; it may be considered the root.

**Analysis** serves as a container for a list of **AnalysisSteps**. This list may initially be empty, but is intended to hold all steps of a given population PK and/or PD analysis. **AnalysisStep** objects will be indexed according to unique identifiers.

#### 4.1.1 AnalysisStep

The **AnalysisStep** is the second-highest level component in PharML. An unrestricted number of components of type **AnalysisStep** are allowed within an **Analysis**.

**AnalysisStep** serves as a container for **Data**, **Model**, **Method**, and **Output** objects. Only one component of each type is allowed within an **AnalysisStep**, but **Data**, **Model** and **Method** objects may be shared between **AnalysisStep** objects.

##### ◆ **AnalysisStepID**

A unique identifier for the **AnalysisStep** object. Mandatory.

##### ◆ **Description**

A text description of the **AnalysisStep**. Optional.

#### 4.1.1.1 Data

The **Data** object is the PharML component that defines the data that will be/has been used to estimate the parameters of the **Model** in a given **AnalysisStep**. A **Data** object consists of sub-components for **DosingHistory**, **Observations**, and **Covariates**.

The **Data** object is intended to be compatible with all modules of the framework, including Design and Interpretation, for which reason different requirements apply to the three sub-components with respect to whether the sub-component should store

data by copy or by reference only, and with respect to whether the sub-component is optional or mandatory.

**NOTE:** Since the storage of data in XML is not particularly efficient with respect to portability, it is envisaged that the subcomponents of **Data** will store references to the data's location, as opposed to the data itself – although this is a matter of implementation. This section has been written on the basis of this assumption.

Data objects may be shared and inherited between **AnalysisStep** objects, and are referenced by unique identifiers.

The structure of **Data** objects will be aligned with appropriate CDISC standards in a later version of PharML Level 1. The following is therefore a preliminary proposal:

◆ **DataID**

A unique identifier for the **Data** object. Mandatory.

◆ **DosingHistory**

A description of the data file containing dosing history, with a specification of type and location, a general definition of units, and a list of all dose events in the dataset. Optional.

➤ **Location**

A text field indicating the location of the data file, either as a physical location expressed as a URL or an SQL statement. One per **DosingHistory** instance. Mandatory.

➤ **Type**

A text field indicating the type of data, e.g. whitespace table (WSTable), comma-separated-value table (CSVTable), SQL, etc. One per **DosingHistory** instance. Mandatory.

➤ **Encoding**

A text field indicating data encoding, e.g. CRLF (DOS/Windows), LF (UNIX/Linux), CR (Apple MacOS). One per **DosingHistory** instance. Mandatory.

➤ **SubjectID**

One per **DosingHistory** instance. Mandatory.

- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
  - **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.
  - **VariableLabel**  
Variable label. Text, optional.
- **OccasionID**  
One per **DosingHistory** instance. Mandatory, if more than one occasion; otherwise optional.
- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
  - **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.
  - **VariableLabel**  
Variable label. Text, optional.
- **DoseAmount**  
One per **DosingHistory** instance. Mandatory.
- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
  - **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.
  - **VariableLabel**  
Variable label. Text, optional.
  - **VariableUnit**

The unit used for dosing amount, e.g. mg. Text to be selected from list (to be defined). Mandatory.

➤ **DoseTime**

One per **DosingHistory** instance. Mandatory.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

▪ **VariableUnit**

The unit used for time of dosing, e.g. h. Text to be selected from list (to be defined). Mandatory.

➤ **DoseDuration**

One per **DosingHistory** instance. Mandatory if infusions used; otherwise optional.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

▪ **VariableUnit**

The unit used for dose duration, e.g. h. Text to be selected from list (to be defined). Mandatory.

➤ **DoseType**

Type of dose. Text, choice of {Bolus, Infusion}. One per **DosingHistory** instance. Mandatory, but may be automatically generated depending on the presence or absence of **DoseDuration**.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **AdministrationRoute**

Route of administration. Text, choice of {IV, IP, IT, IM, SC, PO, TD, etc.}. One per **DosingHistory** instance. Mandatory. This item is important for making a link to dosing compartment, if relevant.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **Formulation**

Type of formulation. Number/text, e.g. {1, 2, 3 etc.}. One per **DosingHistory** instance. Optional.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

- ***VariableLabel***

Variable label. Text, optional.

➤ **RepetitionInterval**

One per **DosingHistory** instance. Optional. Requires **RepetitionNumber** or **SteadyStateIndex**.

- ***VariableID***

Unique variable identifier (to be used in **Model**). Text, mandatory.

- ***VariableIndex***

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

- ***VariableLabel***

Variable label. Text, optional.

- ***VariableUnit***

The unit used for period of time between dose repetitions, e.g. h. Text to be selected from list (to be defined). Mandatory.

➤ **RepetitionNumber**

Number of doses given (incl. the initial dose). Integer. One per **DosingHistory** instance. Optional. Requires **RepetitionInterval**.

- ***VariableID***

Unique variable identifier (to be used in **Model**). Text, mandatory.

- ***VariableIndex***

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

- ***VariableLabel***

Variable label. Text, optional.

➤ **SteadyStateIndex**

Steady state index. Boolean (True/False). One per **DosingHistory** instance. Optional. Requires **RepetitionInterval**.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

◆ **Observations**

A description of the data file containing observations, with a specification of type and location, and a definition of mappings/references for all variables.

➤ **Location**

A text field indicating the location of the data file, either as a physical location expressed as a URL or an SQL statement. One per **Observations** instance. Mandatory.

➤ **Type**

A text field indicating the type of data, e.g. whitespace table (WSTable), comma-separated-value table (CSVTable), SQL, etc. One per **Observations** instance. Mandatory.

➤ **Encoding**

A text field indicating data encoding, e.g. CRLF (DOS/Windows), LF (UNIX/Linux), CR (Apple Mac). One per **Observations** instance. Mandatory.

➤ **SubjectID**

One per **Observations** instance. Mandatory.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

- **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.
  - **VariableLabel**  
Variable label. Text, optional.
- **OccasionID**
- One per **Observations** instance. Mandatory, if more than one occasion; otherwise optional.
- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
  - **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.
  - **VariableLabel**  
Variable label. Text, optional.
- **IndependentVariable**
- Several allowed per **Observations** instance. Mandatory.
- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
  - **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.
  - **VariableLabel**  
Variable label. Text, optional.
  - **VariableUnit**  
Variable unit. Text to be selected from list (to be defined). Mandatory.
  - **VariableType**

Variable type, from {Continuous; Categorical}. Text, mandatory.

- **LOQ**

Limit of quantification for assay. Float, optional.

- **DependentVariable**

Several allowed per **Observations** instance. Mandatory.

- **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

- **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

- **VariableLabel**

Variable label. Text, optional.

- **VariableUnit**

Variable unit. Text to be selected from list (to be defined). Mandatory.

- **VariableType**

Variable type, from {Continuous; Categorical}. Text, mandatory.

- **LOQ**

Limit of quantification for assay. Float, optional.

- ◆ **Covariates**

A description of the data file containing covariates, with a specification of type and location, and a list of covariate values for all subjects in the dataset. Optional.

- **Location**

A text field indicating the location of the data file, either as a physical location expressed as a URL or an SQL statement. One per **Covariates** instance. Mandatory.

- **Type**

A text field indicating the type of data, e.g. whitespace table (WSTable), comma-separated-value table (CSVTable), SQL, etc. One per **Covariates** instance. Mandatory.

➤ **Encoding**

A text field indicating data encoding, e.g. CRLF (DOS/Windows), LF (UNIX/Linux), CR (Apple Mac). One per **Covariates** instance. Mandatory.

➤ **SubjectID**

One per **Covariates** instance. Mandatory.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **OccasionID**

One per **Covariates** instance. Mandatory, if more than one occasion; otherwise optional.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **Covariate**

Several allowed per **Covariates** instance. Mandatory.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

- **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory. May be automatically computed from variable (column) name.

- **VariableLabel**

Variable label. Text, optional.

- **VariableUnit**

Variable unit. Text to be selected from list (to be defined). Mandatory.

#### 4.1.1.2 Model

The **Model** is the PharML component that defines the population PK and/or PD model that will be/has been fitted to the **Data** in a given **AnalysisStep**. Logical expressions and loops will be supported. Mathematical expressions will be encoded using the current MathML 2.0 content standard (Carlisle *et al*, 2003). **ObservationEquations** are mandatory. **AlgebraicEquations** are optional. **DifferentialEquations** are not necessarily mandatory – as an example, they are not required in pure dose-response models.

- ◆ **ParameterModels**

Description of the occasion-within-subject specific parameters used in the model – in terms of fixed effects, covariates and random effects. One per **Model** instance. Mandatory.

- **ParModelElement**

Description of an occasion-within-subject specific parameter. Several allowed per **ParameterModels** instance. Mandatory.

- **VariableID**

Unique variable identifier. Text, mandatory.

- **Code**

Equation code. MathML, mandatory. **NOTE:** random effects to be coded as “EtalIV[x]” for subject-level random effects, and as “EtalOV[x]” for occasion-within-subject-level random effects, where [x] is an integer, e.g. EtalIV1, EtalOV2.

### ◆ **AlgebraicEquations**

Description of the algebraic equations used in the model. One per **Model** instance. Optional.

#### ➤ **AEVectorElement**

Description of the right-hand side of an algebraic equation. Several allowed per **AlgebraicEquations** instance. Mandatory.

- **VariableID**  
Unique variable identifier. Text, mandatory.
- **Code**  
Equation code. MathML, mandatory.

### ◆ **DifferentialEquations**

Description of the differential equations used in the model. One per **Model** instance. Optional.

#### ➤ **RHSVectorGElement**

Description of a right-hand side element for a differential equation (ODE part). Several allowed per **DifferentialEquations** instance.

- **ElementID**  
Unique element identifier (x). This is always an element in a vector and may be a compartment number. Integer, mandatory.
- **Code**  
Equation code. MathML, mandatory.

#### ➤ **RHSMatrixSigmaElement**

Description of a right-hand side element for a differential equation (additional SDE part). Several allowed per **DifferentialEquations** instance, but no more than the number of **RHSvectorGElement** instances squared. Optional.

- **ElementID**  
Unique element identifier (x.y). This is always an element in a matrix. Integer, mandatory.
- **Code**  
Equation code. MathML, mandatory.

#### ➤ **InitialCondition**

Description of an initial condition for a differential equation. Several allowed per **DifferentialEquations** instance.

- **ElementID**

Unique element identifier (x). This is always an element in a vector and may be a compartment number. Integer, mandatory.

- **Code**

Equation code. MathML, mandatory.

◆ **ObservationEquations**

Description of the observation equations used in the model. One per **Model** instance. Mandatory.

- **ObsVectorElement**

Description of the right-hand side of an observation equation. Several allowed per **ObservationEquations** instance. Mandatory.

- **VariableID**

Unique variable identifier (to use as link to **Data**). Text, mandatory.

- **Code**

Equation code for predictions (e.g. concentrations for continuous data, a probability for categorical data). MathML, mandatory.

- **LinkFunction**

Predefined function, choice of {Identity, Inverse, InverseSquared, Log, Logit}. Text, mandatory if **VariableID** is defined as categorical.

◆ **Distributions**

Description of the distributions of the various random effects of the model. One per **Model** instance. Mandatory.

- **EtaDistribution**

Description of the distribution of a random effect for inter-individual or inter-occasion-within-subject variability. Several allowed per **Distributions** instance. Mandatory.

- **EtaID**

Unique identifier for the random effect. Must match that used in the code of the **ParameterModels**.

- ***DistributionType***

One of {ParametricNormal; ParametricOther; Nonparametric; Truncated}. Depending on the option chosen, **EtaDistributionParameters** describing the distribution may be automatically generated. Text, mandatory.

- **EpsilonDistribution**

Description of the distribution of a random effect for residual variability. Several allowed per **Distributions** instance. Mandatory.

- ***EpsilonID***

Unique identifier for the random effect. Must match that used in the code of **ObservationEquations**.

- ***DistributionType***

One of {ParametricNormal; ParametricOther; Nonparametric; Truncated}. Depending on the option chosen, **EpsilonDistributionParameters** describing the distribution may be automatically generated. Text, mandatory.

- ◆ **Mixture**

Mixture models can be applied to describe different distributions for parameters or submodels. Multiple allowed per **Model** instance. Optional.

- **MixtureID**

Unique identifier for the mixture. Must match that used in code elsewhere. Text, mandatory.

- **Subpopulation**

Subpopulation included in the mixture. Multiple allowed per **Mixture** instance.

- ***SubpopulationID***

Unique identifier for the mixture subpopulation. Text, mandatory.

- ***ParameterID***

Fraction of the population made up by the subpopulation. Mandatory.  $n_s-1$  **ParameterIDs** must be defined for the **Mixture** object, where  $n_s$  is the number of subpopulations.

- **SubpopulationModel**

Code for computing the fraction of the population in the subpopulation. MathML.

- ◆ **Parameters**

Description of all of the parameters of a **Model** that can potentially be estimated within an **AnalysisStep**. One per **Model** instance. Mandatory.

- **FixedEffectsParameter**

Description of a fixed effects parameter. Several allowed per **Parameters** instance. Mandatory.

- **ParameterID**

Unique identifier for parameter. Must match that used in the code of **ParameterModels**, **AlgebraicEquations**, **DifferentialEquations**, **ObservationsEquations**, **Mixtures**, if used there. Text, mandatory.

- **Label**

Label for parameter. Text, optional.

- **Unit**

Unit for parameter. Text, mandatory.

- **InitialEstimate**

Initial estimate for parameter. Float, optional.

- **LowerBound**

Lower bound for parameter. Float, optional.

- **UpperBound**

Upper bound for parameter. Float, optional.

- **Fixed**

Is the parameter fixed? Boolean (True/False), mandatory.

- **PriorMean**

Mean of the prior of the parameter (assuming a normal distribution for the prior). Float. Optional, but, if defined, requires that **PriorSD** is also defined.

- **PriorSD**

Standard deviation of the prior of the parameter (assuming a normal distribution for the prior). Float. Optional, but, if defined, requires that **PriorMean** is also defined.

- **PriorCorrelation**

Prior correlation between the parameter and another parameter, for which a (normally distributed) prior is also defined. Optional, but, if defined, requires identical **PriorCorrelation** definition for the other parameter.

- **OtherID**  
**ParameterID** for the other parameter. Text, mandatory.
- **CorrelationValue**  
Value of the correlation coefficient. Float, mandatory.

- **EtaDistributionParameter**

Description of a parameter of the distribution of the inter-individual or inter-occasion-within-subject random effects. Several allowed per **Parameters** instance. Optional.

- **ParameterID**

Unique identifier for parameter. Must match the choice made in **Distributions** (for normal distributions, use “OmegallV[x.y]” for covariance matrix elements for subject-level random effects, and “OmegalOV[x.y]” for covariance matrix elements for occasion-within-subject-level random effects). Text, mandatory.

- **Label**

Label for parameter. Text, optional.

- **Unit**

Unit for parameter. Text, optional.

- **InitialEstimate**

Initial estimate for parameter. Float, mandatory.

- **LowerBound**

Lower bound for parameter. Float, optional.

- **UpperBound**

Upper bound for parameter. Float, optional.

- **Fixed**  
Is the parameter fixed? Boolean (True/False), mandatory.
  - **PriorMean**  
Mean of the prior of the parameter (assuming a normal distribution for the prior). Float. Optional, but, if defined, requires that **PriorSD** is also defined. Cannot be defined, if **PriorMode** is also defined.
  - **PriorSD**  
Standard deviation of the prior of the parameter (assuming a normal distribution for the prior). Float. Optional, but, if defined, requires that **PriorMean** is also defined. Cannot be defined, if **PriorMode** is also defined.
  - **PriorCorrelation**  
Prior correlation between the parameter and another parameter, for which a (normally distributed) prior is also defined. Optional, but, if defined, requires identical **PriorCorrelation** definition for the other parameter. Cannot be defined, if **PriorMode** is also defined.
    - **OtherID**  
**ParameterID** for the other parameter. Text, mandatory.
    - **CorrelationValue**  
Value of the correlation coefficient. Float, mandatory.
  - **PriorMode**  
Mode of the prior of the parameter (assuming an inverse Wishart distribution for the prior). Float. Optional, but, if defined, requires that **PriorDOF** is also defined. Cannot be defined, if **PriorMean** is also defined.
  - **PriorDOF**  
Degrees of freedom for the prior of the parameter (assuming an inverse Wishart distribution for the prior). Float. Optional, but, if defined, requires that **PriorMode** is also defined. Cannot be defined, if **PriorMean** is also defined.
- **EpsilonDistributionParameter**  
Description of a parameter of the distribution of the residual error random effects. Several allowed per **Parameters** instance. Mandatory.

- **ParameterID**  
Unique identifier for parameter. Must match the choice made in **Distributions** (for normal distributions, use “Sigma[x.y]” for covariance matrix elements). Text, mandatory.
- **Label**  
Label for parameter. Text, optional.
- **Unit**  
Unit for parameter. Text, optional.
- **InitialEstimate**  
Initial estimate for parameter. Float, mandatory.
- **LowerBound**  
Lower bound for parameter. Float, optional.
- **UpperBound**  
Upper bound for parameter. Float, optional.
- **Fixed**  
Is the parameter fixed? Boolean (True/False), mandatory.
- **PriorMean**  
Mean of the prior of the parameter (assuming a normal distribution for the prior). Float. Optional, but, if defined, requires that **PriorSD** is also defined. Cannot be defined, if **PriorMode** is also defined.
- **PriorSD**  
Standard deviation of the prior of the parameter (assuming a normal distribution for the prior). Float. Optional, but, if defined, requires that **PriorMean** is also defined. Cannot be defined, if **PriorMode** is also defined.
- **PriorCorrelation**  
Prior correlation between the parameter and another parameter, for which a (normally distributed) prior is also defined. Optional, but, if defined, requires identical **PriorCorrelation** definition for the other parameter. Cannot be defined, if **PriorMode** is also defined.
  - **OtherID**

**ParameterID** for the other parameter. Text, mandatory.

- **CorrelationValue**  
Value of the correlation coefficient. Float, mandatory.
- **PriorMode**  
Mode of the prior of the parameter (assuming an inverse Wishart distribution for the prior). Float. Optional, but, if defined, requires that **PriorDOF** is also defined. Cannot be defined, if **PriorMean** is also defined.
- **PriorDOF**  
Degrees of freedom for the prior of the parameter (assuming an inverse Wishart distribution for the prior). Float. Optional, but, if defined, requires that **PriorMode** is also defined. Cannot be defined, if **PriorMean** is also defined.
- **FinalEstimate**  
Final estimate for parameter. Float, optional.
- **StandardError**  
Precision of the final estimate for parameter. Float, optional.

#### 4.1.1.3 Method

The **Method** is the PharML component that defines the estimation method that will be/has been used to fit the **Model** to the **Data** in a given **AnalysisStep**.

◆ **SoftwareImplementation**

The software in which the method is implemented. Text, mandatory.

◆ **SoftwareVersion**

The version of the software implementing the method. Text, mandatory.

◆ **SoftwarePath**

The path to the software implementing the method. Text URL, mandatory.

◆ **ExecutableCommand**

The command executed to run the model. Text, optional (software-specific).

◆ **SoftwareSpecificData**

TBD for all possible options: NONMEM, MONOLIX, BUGS, S-PLUS, R, MATLAB, S-ADAPT, SAS, ... Optional.

◆ **Algorithm**

The algorithm used for parameter estimation. Text, choice of {FO; FOCE; Laplacian; FOCEI; SAEM; ...}. Mandatory.

◆ **Seed**

The random seed used for parameter estimation. Text, dependent on **Algorithm** and **SoftwareImplementation**.

#### 4.1.1.4 Output

The **Output** is the PharML component that defines the standard output required to describe the model and how well it fits the data as well as the convergence of the estimation algorithm, when the **Model** has been fitted to the **Data** using the **Method** in a given **AnalysisStep**.

◆ **Parameters**

Description of all of the parameters of a **Model** that have been estimated within an **AnalysisStep**. One per **Output** instance. Mandatory.

➤ **FixedEffectsParameter**

Description of a fixed effects parameter. Several allowed per **Parameters** instance. Mandatory.

▪ **ParameterID**

Unique identifier for parameter, mapping to entry in **Model** -> **FixedEffectsParameter**. Text, mandatory.

▪ **FinalEstimate**

Final estimate for parameter. Float, mandatory.

▪ **StandardError**

Precision of the final estimate for parameter. Float, optional.

➤ **EtaDistributionParameter**

Description of a parameter of the distribution of the inter-individual or inter-occasion-within-subject random effects. Several allowed per **Parameters** instance. Optional.

▪ **ParameterID**

Unique identifier for parameter, mapping to entry in **Model** -> **EtaDistributionParameter**. Text, mandatory.

▪ **FinalEstimate**

Final estimate for parameter. Float, mandatory.

- **StandardError**

Precision of the final estimate for parameter. Float, optional.

- **EpsilonDistributionParameter**

Description of a parameter of the distribution of the residual error random effects. Several allowed per **Parameters** instance. Mandatory.

- **ParameterID**

Unique identifier for parameter, mapping to entry in **Model** -> **EpsilonDistributionParameter**. Text, mandatory.

- **FinalEstimate**

Final estimate for parameter. Float, mandatory.

- **StandardError**

Precision of the final estimate for parameter. Float, optional.

- ◆ **RunTimeEnvironment**

Information describing the runtime environment in which the modelling activity was carried out.

- **CPUType**

CPU make, model and speed. Text, mandatory.

- **OperatingSystemType**

Operating system type, e.g. UNIX, Linux, Windows, or DOS. Text, mandatory.

- **OperatingSystemVersion**

Version of operating system. Text, mandatory.

- **RAM**

The amount of random access memory available to the system. Integer, Gb. Optional.

- **StartTime**

Full date and time of **AnalysisStep** start. Timestamp, optional.

- **EndTime**

Full date and time of **AnalysisStep** termination. Timestamp, optional.

- ◆ **SoftwareSpecificData**

Will include product-specific data items relating to NONMEM, MONOLIX, BUGS, S-PLUS, R, MATLAB, S-ADAPT, SAS, ... (e.g. this will include objective function or likelihood values depending on the application). To be further detailed.

◆ **FitOutput**

A description of the data file containing model fit-related output, with a specification of type and location, and a definition of mappings/references for all variables.

- **Location**  
A text field indicating the location of the data file, either as a physical location expressed as a URL or an SQL statement. One per **FitOutput** instance. Mandatory.
- **Type**  
A text field indicating the type of data, e.g. whitespace table (WSTable), comma-separated-value table (CSVTable), SQL, etc. One per **FitOutput** instance. Mandatory.
- **Encoding**  
A text field indicating data encoding, e.g. CRLF (DOS/Windows), LF (UNIX/Linux), CR (Apple Mac). One per **FitOutput** instance. Mandatory.
- **SubjectID**  
One per **FitOutput** instance. Mandatory.
  - **VariableID**  
Unique variable identifier (as used in **Model**). Text, mandatory.
  - **VariableIndex**  
Variable (column) index in the input data file. Integer, mandatory.
  - **VariableLabel**  
Variable label. Text, optional.
- **OccasionID**  
One per **FitOutput** instance. Mandatory, if more than one occasion; otherwise optional.
  - **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

- ***VariableIndex***

Variable (column) index in the input data file. Integer, mandatory.

- ***VariableLabel***

Variable label. Text, optional.

➤ **IndependentVariable**

Several allowed per **FitOutput** instance. Mandatory.

- ***VariableID***

Unique variable identifier (to be used in **Model**). Text, mandatory.

- ***VariableIndex***

Variable (column) index in the input data file. Integer, mandatory.

- ***VariableLabel***

Variable label. Text, optional.

- ***VariableUnit***

Variable unit. Text, mandatory.

- ***VariableType***

Variable type, from {Continuous; Categorical}. Text, mandatory.

➤ **PopulationPrediction**

Several allowed per **FitOutput** instance. Optional.

- ***VariableID***

Unique variable identifier (to be used in **Model**). Text, mandatory.

- ***VariableIndex***

Variable (column) index in the output data file. Integer, mandatory.

- ***VariableLabel***

Variable label. Text, optional.

- **VariableUnit**  
Variable unit. Text, mandatory.
- **VariableType**  
Variable type, from {Continuous; Categorical}. Text, mandatory.

➤ **IndividualPrediction**

Several allowed per **FitOutput** instance. Optional.

- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
- **VariableIndex**  
Variable (column) index in the output data file. Integer, mandatory.
- **VariableLabel**  
Variable label. Text, optional.
- **VariableUnit**  
Variable unit. Text, mandatory.
- **VariableType**  
Variable type, from {Continuous; Categorical}. Text, mandatory.

➤ **Residual**

Several allowed per **FitOutput** instance. Optional.

- **VariableID**  
Unique variable identifier (to be used in **Model**). Text, mandatory.
- **VariableIndex**  
Variable (column) index in the output data file. Integer, mandatory.
- **VariableLabel**  
Variable label. Text, optional.
- **VariableUnit**  
Variable unit. Text, mandatory.

➤ **WeightedResidual**

Several allowed per **FitOutput** instance. Optional.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the output data file. Integer, mandatory.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **IndividualWeightedResidual**

Several allowed per **FitOutput** instance. Optional.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the output data file. Integer, mandatory.

▪ **VariableLabel**

Variable label. Text, optional.

◆ **EmpiricalBayesEstimates**

A description of the data structure or file containing individualized parameter estimates, with a specification of type and location, and a definition of mappings/references for all variables.

➤ **Location**

A text field indicating the location of the data file, either as a physical location expressed as a URL or an SQL statement. One per **EmpiricalBayesEstimates** instance. Optional.

➤ **Type**

A text field indicating the type of data, e.g. whitespace table (WSTable), comma-separated-value table (CSVTable), SQL, etc. One per **EmpiricalBayesEstimates** instance. Optional.

➤ **Encoding**

A text field indicating data encoding, e.g. CRLF (DOS/Windows), LF (UNIX/Linux), CR (Apple Mac). One per **EmpiricalBayesEstimates** instance. Optional.

➤ **SubjectID**

One per **EmpiricalBayesEstimates** instance. Mandatory.

▪ **VariableID**

Unique variable identifier (as used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **OccasionID**

One per **EmpiricalBayesEstimates** instance. Mandatory, if more than one occasion; otherwise optional.

▪ **VariableID**

Unique variable identifier (to be used in **Model**). Text, mandatory.

▪ **VariableIndex**

Variable (column) index in the input data file. Integer, mandatory.

▪ **VariableLabel**

Variable label. Text, optional.

➤ **ParameterValue**

Empirical Bayes estimate of a parameter. Multiple allowed per **EmpiricalBayesEstimates** instance. Mandatory.

▪ **ParameterID**

Unique identifier for parameter, mapping to entry in **Model** -> **ParameterModels**. Text, mandatory.

▪ **EBE**

Individualized estimate of the parameter. Float, mandatory.

## 5 Perspectives

### 5.1 Future enhancements to PharML: Level 2 and beyond

- CDISC integration with respect to **Data** objects will be pursued.
- Extension to Design and Interpretation sections of the NLMEc URS.
- Examples of schemas and implementations of models will be released.

### 5.2 Relationship to other efforts

PharML is related structurally to SBML (Systems Biology Markup Language, <http://www.sbml.org/>) and utilizes existing CDISC (<http://www.cdisc.org/>) and MathML (<http://www.w3.org/Math>).

### 5.3 Availability

The full text of these specifications will be made available online once completed.

## 6 References

Carlisle D, Ion P, Miner R, Poppelier N (ed.) (2003). *Mathematical Markup Language (MathML) Version 2.0 (Second Edition) – W3 Recommendation 21 October 2003*. W3 Consortium, <http://www.w3.org/TR/2003/REC-MathML2-20031021/>

Frame, B (2007). Mixture Modelling with NONMEM V in: Ette EI, Williams PJ. *Pharmacometrics: The Science of Quantitative Pharmacology*. Wiley, New Jersey, 723-757.